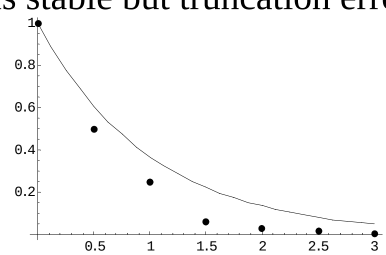
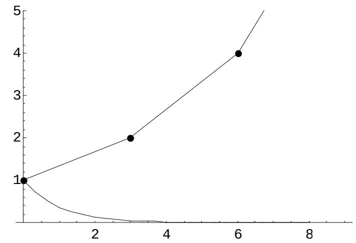
CS205 - Class 16

Readings: 9.3
Covered in Class: 1, 2, 3, 4, 5, 6

ODE's (Continued)

1. Recall the model ODE from last time. $y' = f(t, y)$
    a. We stated that $\lambda > 0$ is ill-posed. But why?
        i. Errors accumulated and they increase exponentially.
2. (Forward) **Euler's Method** $\frac{y_{k+1} - y_k}{h} = f(t_k, y_k)$ or $y_{k+1} = y_k + hf(t_k, y_k)$
    a. **Accuracy**, truncation error usually dominates round-off error in ODE's.
    b. **Local truncation error** $y_{k+1} = y_k + hf(t_k, y_k) + O(h^2)$
        i. y_{k+1} is calculated by ignoring the $O(h^2)$ term.
        ii. If y_0 is exact, the error in y_1 is $O(h^2)$.
    c. **Global truncation error** integrating from $t = t_o$ to $t = t_{final}$ with $n = O(1/h)$ steps gives a total error of $O(nh^2) = O(h)$.
        i. Euler's method is *1st order accurate* with $\frac{y_{k+1} - y_k}{h} = f(t_k, y_k) + O(h)$
    d. For **stability** consider the model equation $y' = \lambda y$ where $\lambda < 0$
        i. For a general ode λ is df/dy or an eigenvalue of the Jacobian matrix
        ii. Euler's method applied to the model equation is $y_{k+1} = y_k + h\lambda y_k = (1 + h\lambda)y_k$
        iii. So $y_k = (1 + h\lambda)^k y_o$ and the error shrinks when $|1 + h\lambda| < 1$
            1. Thus, $-2 < h\lambda < 0$ is needed for stability
    e. Example
        i. Forward Euler on $y' = -y$ for $y_0 = 1$, $t_0 = 0$. Stability is $h<2$
        ii. h=.5 is stable but truncation errors cause it to be smaller
            1. 

        iii. Same example but with *h=3* we get unstable

1. 

3. As an aside, stability restriction related to your ability to get accuracy. Consider a large lambda and you also have another eigenvalue that is smaller. You need a small time step for the large eigenvalue. For example if you had y = c1 y1 + c2 y2. If they differ by a lot you get a stiff problem. For stiff problems you want a method with no stability requirement.
4. Backward (Implicit) Euler $\frac{y_{k+1}-y_k}{h} = f(t_{k+1}, y_{k+1})$
    a. 1st order accurate
    b. Backward Euler applied to the model equation $y' = \lambda y$ is $y_{k+1} = y_k + h\lambda y_{k+1}$
        i. So $y_{k+1} = (1-h\lambda)^{-1} y_k$ and $y_k = (1-h\lambda)^{-k} y_o$
        ii. The error shrinks when $|(1-h\lambda)^{-1}| < 1$
        iii. Thus, $-\infty < h\lambda < 0$ is needed for stability
        iv. i.e. stable for all h or unconditionally stable
    c. Generally need to solve a nonlinear equation to find y_{k+1}
        i. Can use Newton iteration, i.e. linearize, solve, linearize, solve, etc.
        ii. For some applications, only one linearize and solve cycle is used
    d. One can take very large time steps since it is stable
        i. However it is not very accurate
        ii. As $h \to \infty$, $y_{k+1} = y_k + h\lambda y_{k+1} \to 0 = 0 + h\lambda y_{k+1}$ or $y_{k+1} = 0$
        iii. This is the long run solution for $y' = \lambda y$ with $\lambda < 0$, but we get there too fast
        iv. Everything damps out too quick, i.e. not accurate
5. **Trapezoidal rule** $\frac{y_{k+1}-y_k}{h} = \frac{f(t_k, y_k) + f(t_{k+1}, y_{k+1})}{2}$
    a. 2nd order accurate
    b. Trapezoidal rule applied to the model equation is $y_{k+1} = y_k + \frac{h\lambda}{2}\left(y_k + y_{k+1}\right)$
        i. So $y_{k+1} = (1+h\lambda/2)/(1-h\lambda/2) y_k$ and $y_k = (1+h\lambda/2)^k/(1-h\lambda/2)^k y_o$
        ii. The error shrinks when $|(1+h\lambda/2)/(1-h\lambda/2)| < 1$
        iii. Thus, $-\infty < h\lambda < 0$ is needed for stability
        iv. i.e. unconditionally stable
    c. Generally need to solve a nonlinear equation to find y_{k+1}
        i. One can take very large time steps since it is stable
        ii. However it is not very accurate

iii. As $h \to \infty$, $y_{k+1} = y_k + \frac{h\lambda}{2}\left(y_k + y_{k+1}\right) \to 0 = 0 + \frac{h\lambda}{2}\left(y_k + y_{k+1}\right)$ or $y_{k+1} = -y_k$

iv. This is NOT the long time solution for $y' = \lambda y$

v. Bad oscillatory behavior

6. **1st order Runge-Kutta** is Euler's method $\frac{y_{k+1} - y_k}{h} = f(t_k, y_k)$

7. **2nd order Runge-Kutta** $\frac{y_{k+1} - y_k}{h} = \frac{k_1 + k_2}{2}$

   a. $k_1 = f(t_k, y_k)$ and $k_2 = f(t_k + h, y_k + hk_1)$

8. **4th order Runge-Kutta** $\frac{y_{k+1} - y_k}{h} = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$

   a. $k_1 = f(t_k, y_k)$, $k_2 = f(t_k + h/2, y_k + hk_1/2)$, $k_3 = f(t_k + h/2, y_k + hk_2/2)$ and $k_4 = f(t_k + h, y_k + hk_3)$

9. **TVD Runge Kutta**

   a. 1st order accurate TVD RK is Euler's method

   b. 2nd order accurate TVD RK is the standard second order accurate RK scheme

      i. Also known as the midpoint rule, the modified Euler method, and Heun's predictor-corrector method

      ii. Take two successive forward Euler steps

         1. $\frac{y_{k+1} - y_k}{h} = f(t_k, y_k)$ and $\frac{y_{k+2} - y_{k+1}}{h} = f(t_{k+1}, y_{k+1})$

      iii. Average the initial and final state

         1. $y_{k+1} = \frac{1}{2} y_k + \frac{1}{2} y_{k+2}$

      iv. Same as above, but here one can see the averaging at work

      v. If the solution is well behaved for each Euler step, then since linear interpolation is well behaved, the result is well behaved

   c. 3rd order accurate TVD RK

      i. Take two successive forward Euler steps

         1. $\frac{y_{k+1} - y_k}{h} = f(t_k, y_k)$ and $\frac{y_{k+2} - y_{k+1}}{h} = f(t_{k+1}, y_{k+1})$

      ii. Average the initial and final state

         1. $y_{k+1/2} = \frac{3}{4} y_k + \frac{1}{4} y_{k+2}$

      iii. Take another Euler step

         1. $\frac{y_{k+3/2} - y_{k+1/2}}{h} = f(t_{k+1/2}, y_{k+1/2})$

      iv. Then average yet again

         1. $y_{k+1} = \frac{1}{3} y_k + \frac{2}{3} y_{k+3/2}$